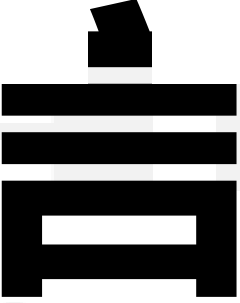
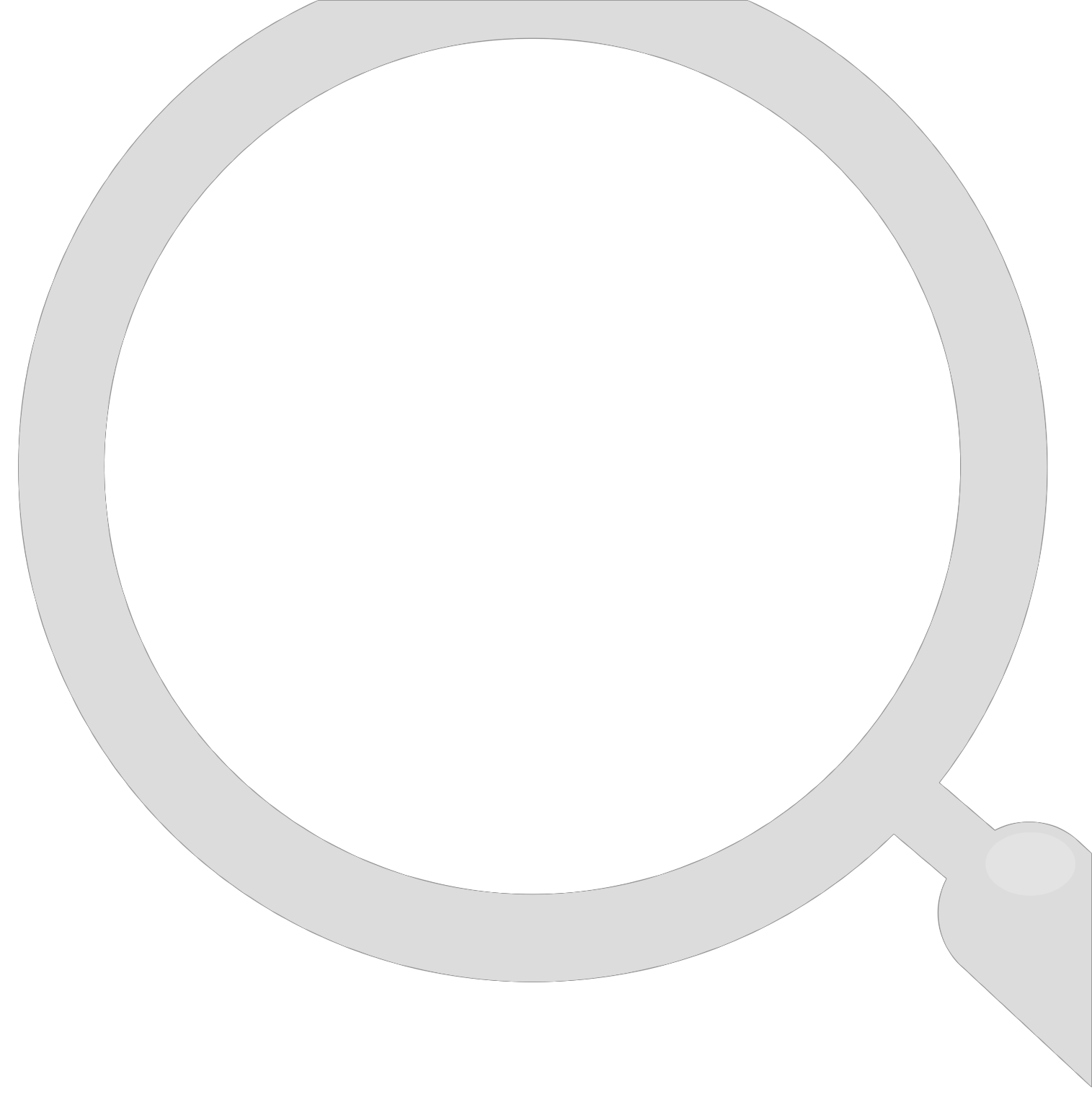
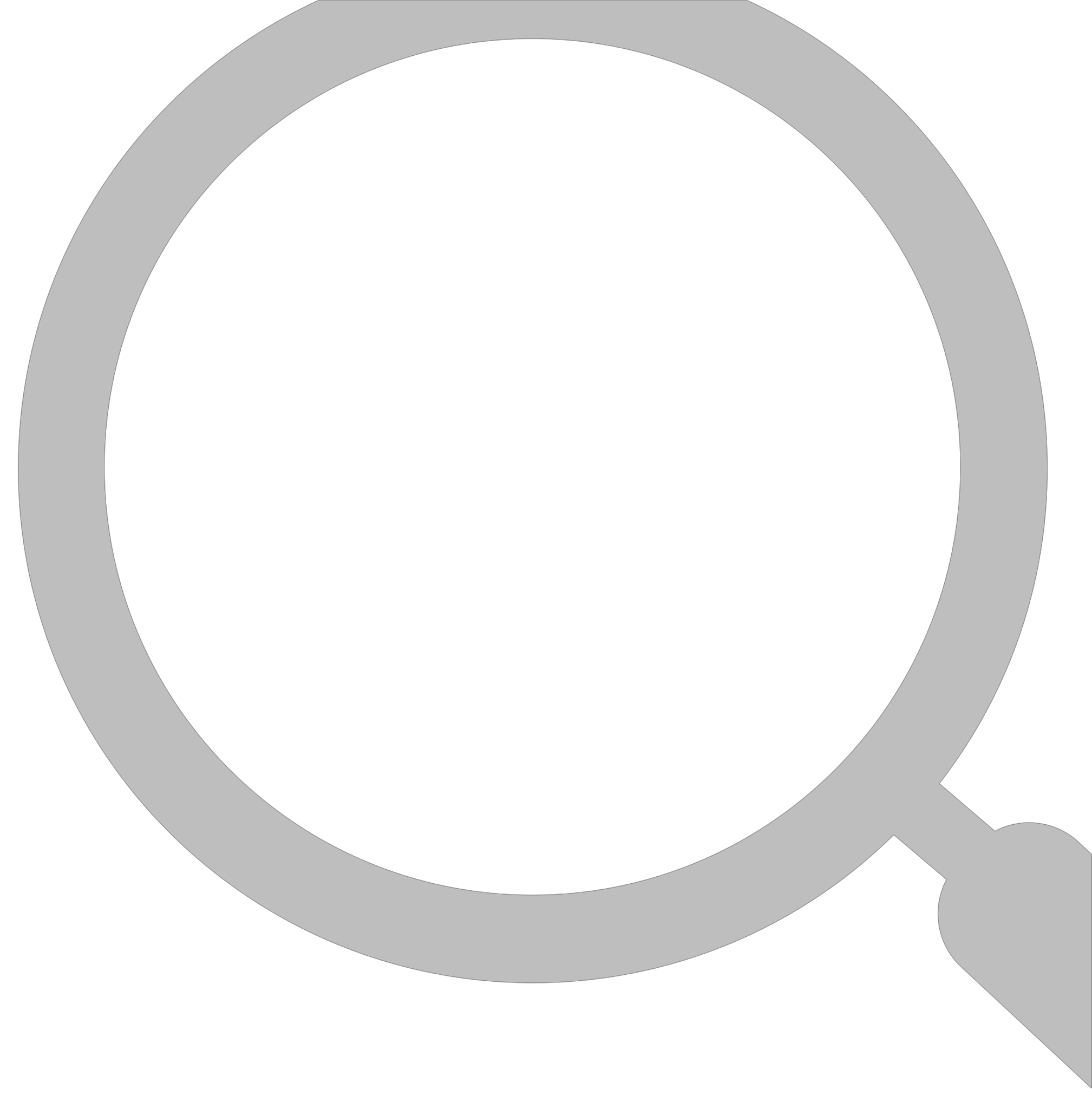
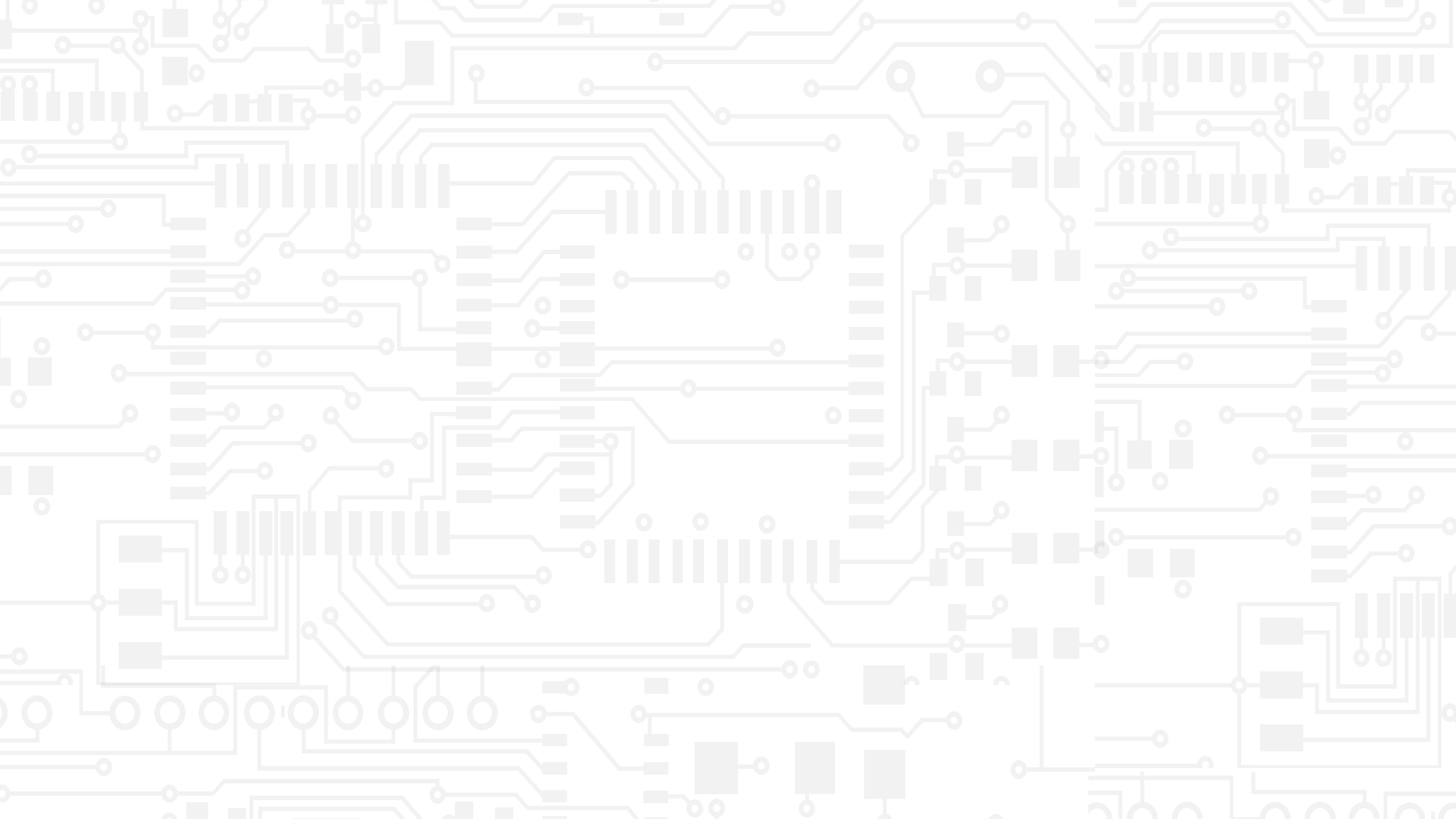
**Python快速入门**

## 嵩 天





嵩夭

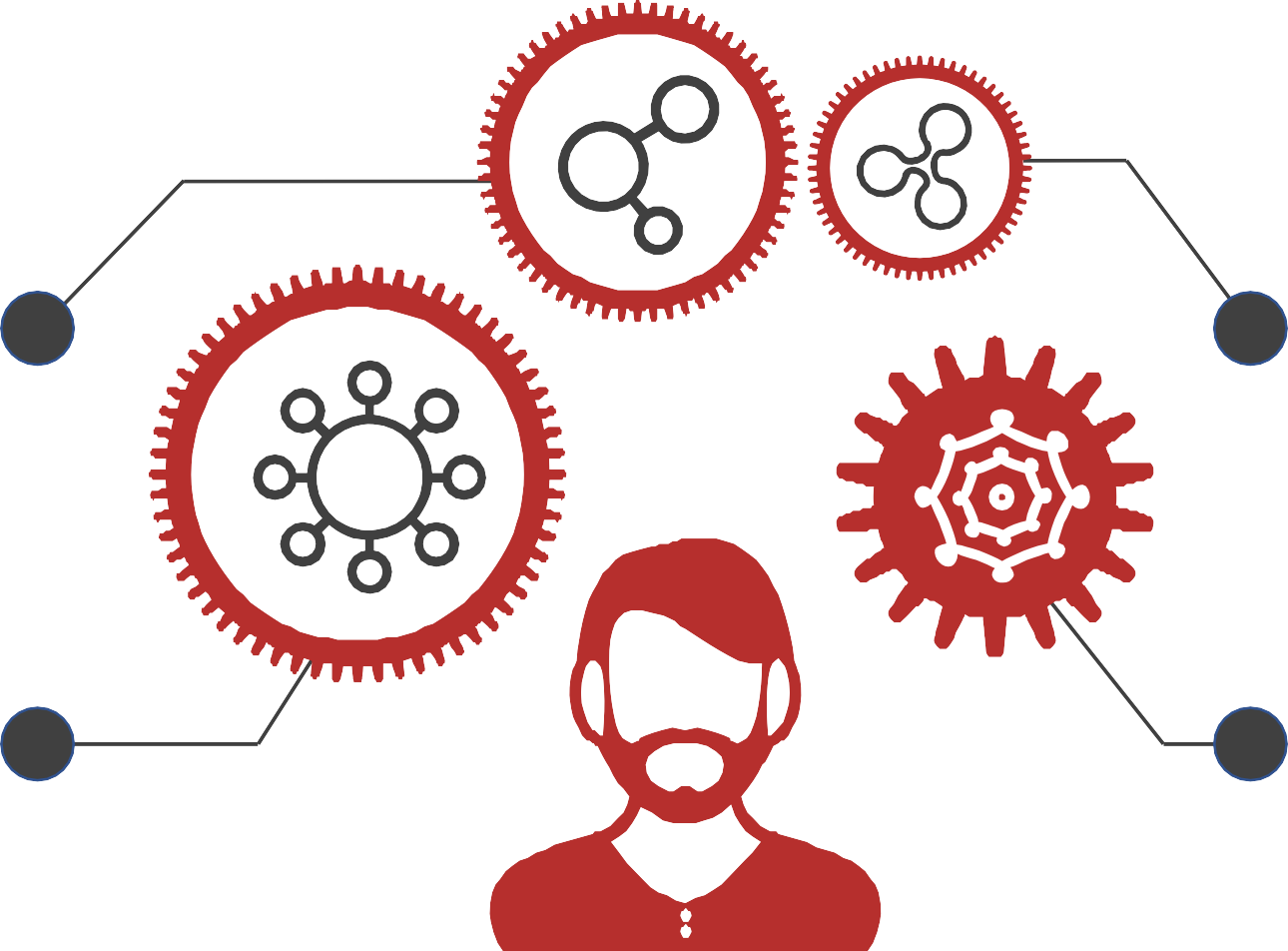


#### Python快速入门

**单元开篇**

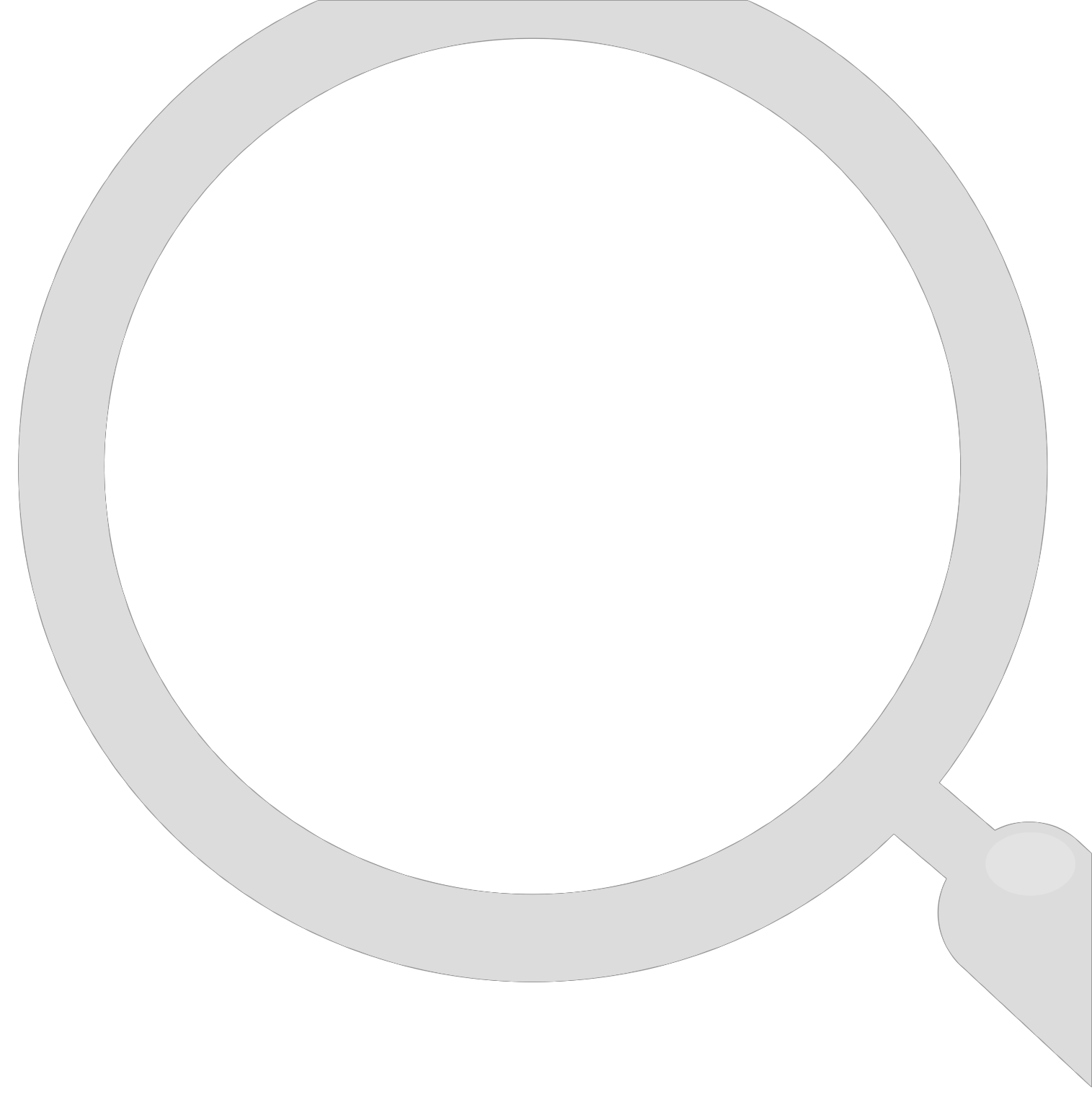
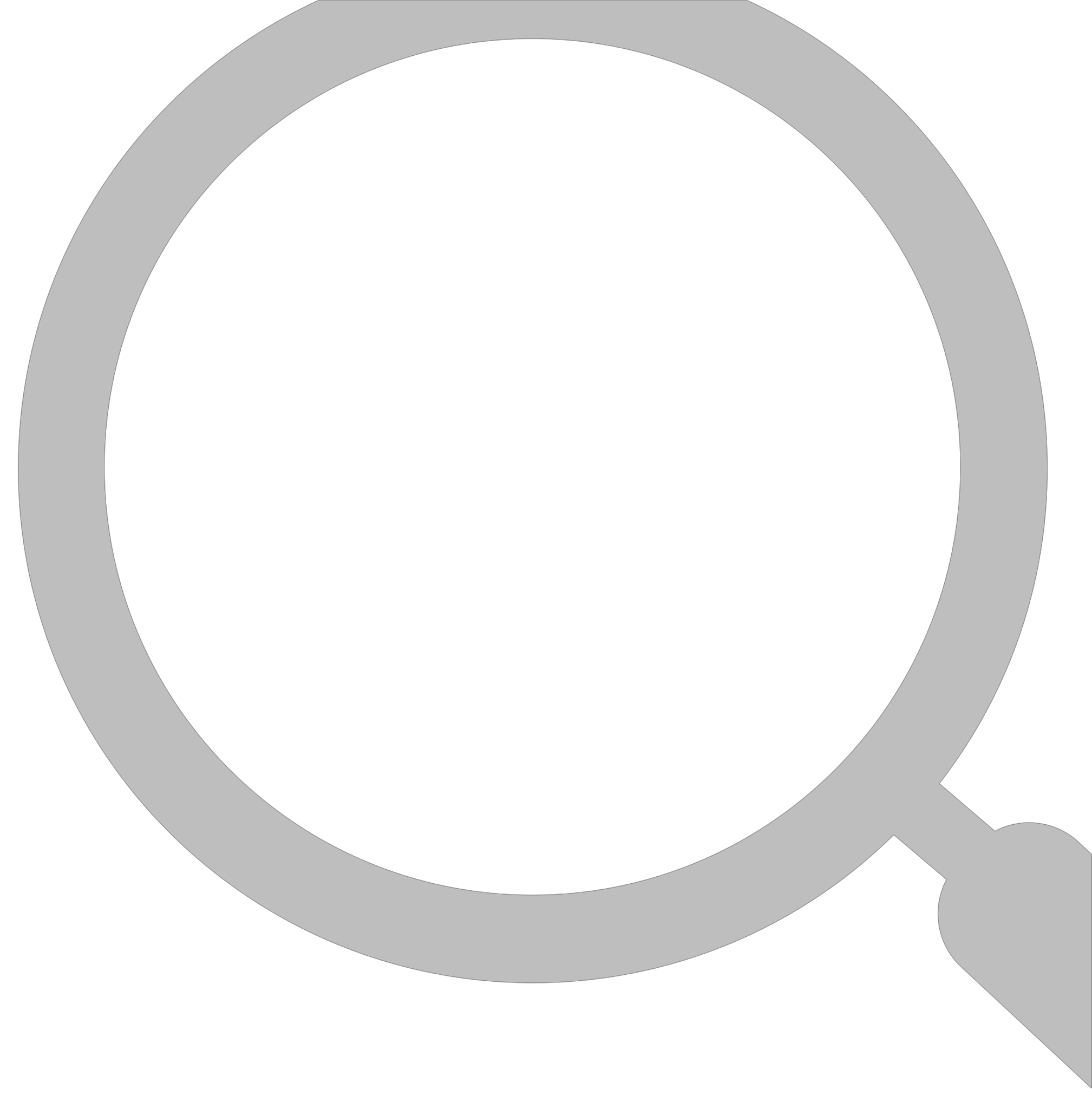
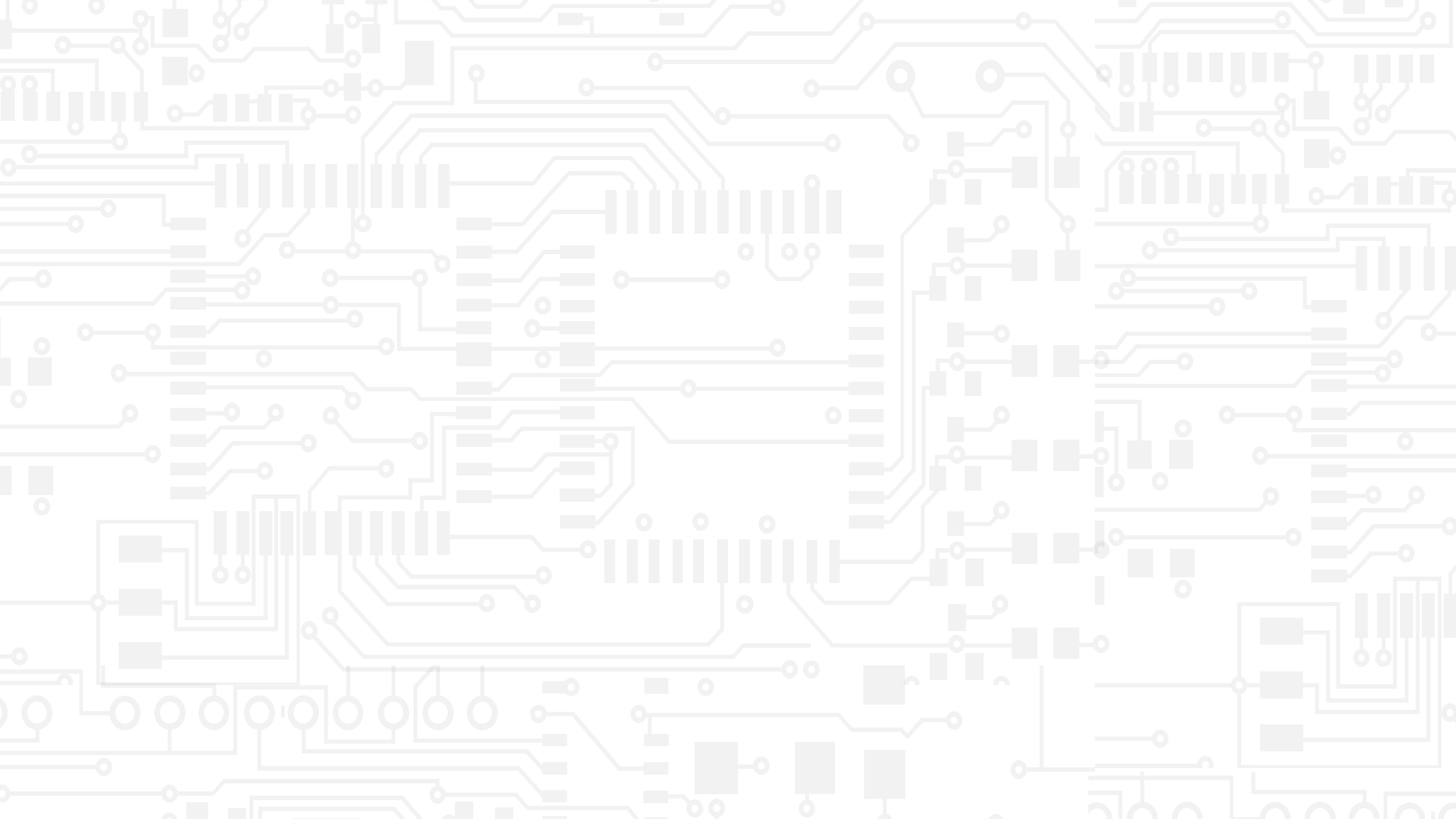
单元开篇

**(2) 编译与解释**

1. **演进中的编程语言**
   1. **Python语言特点**
   2. **计算生态与超级语言**



## 不容错过的Python语言概述



#### Python快速入门

**演进中的编程语言**

Basic, C, C++, C#, CSS, Fortran, Go, HTML, Java,

JavaScript, Lisp, Lua, VC++, Object C, Pascal, Perl, PHP, PostScript, Python, Ruby, Scala, SQL, Swift, VBA,

VB.NET, Verilog, VHDL, Visual Basic… 编程语言，也是一个江湖！

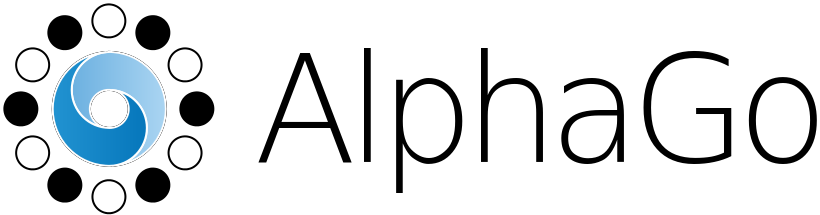
**计算机系统结构时代**

**网络和视窗时代**

**复杂信息系统时代**

**人工智能时代**

1946-1981 1981-2008 2008-2016 2016至今



### 计算机系统结构时代

**网络和视窗时代**

**复杂信息系统时代**

**人工智能时代**

1946-1981 1981-2008 2008-2016 2016至今

### 计算性能问题

**计算交互问题**

**数据周期问题**

**人类需求问题**

**计算机系统结构时代**

**网络和视窗时代**

**复杂信息系统时代**

**人工智能时代**

1946-1981 1981-2008 2008-2016 2016至今

C语言 Java

HTML/VC/ASP

JS/Java C++/PHP/R…

Python

…

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **编程语言** | **学习内容** | **语言本质** | **解决问题** | **适用岗位** |
| C | 指针、内存、数据类型 | 理解计算机系统结构 | 性能 | 系统底层开发 |
| Java | 对象、跨平台、运行时 | 理解主客体关系 | 跨平台 | 网络后台及APP开发 |
| C++ | 对象、多态、继承 | 理解主客体关系 | 大规模程序 | 应用类特定方向 |
| VB/VC | 对象、按钮、文本框 | 理解交互逻辑 | 桌面交互 | 不确定 |
| Python | 编程逻辑、第三方库 | 理解问题求解 | 各类问题 | 大数据/人工智能等 |

## 各编程语言使命不同，Python是计算时代演进的选择！

* 计算机性能已经不是解决一般问题的瓶颈
* 大数据、云计算、物联网、信息安全、人工智能等需求爆发
* 计算领域核心矛盾转化为产量、效率和方法的不足
* 在2018年及以后，面对应用导向计算需求，该使用什么编程语言？

Python是通用语言

1

Python是脚本语言

2

Python是胶水语言

3

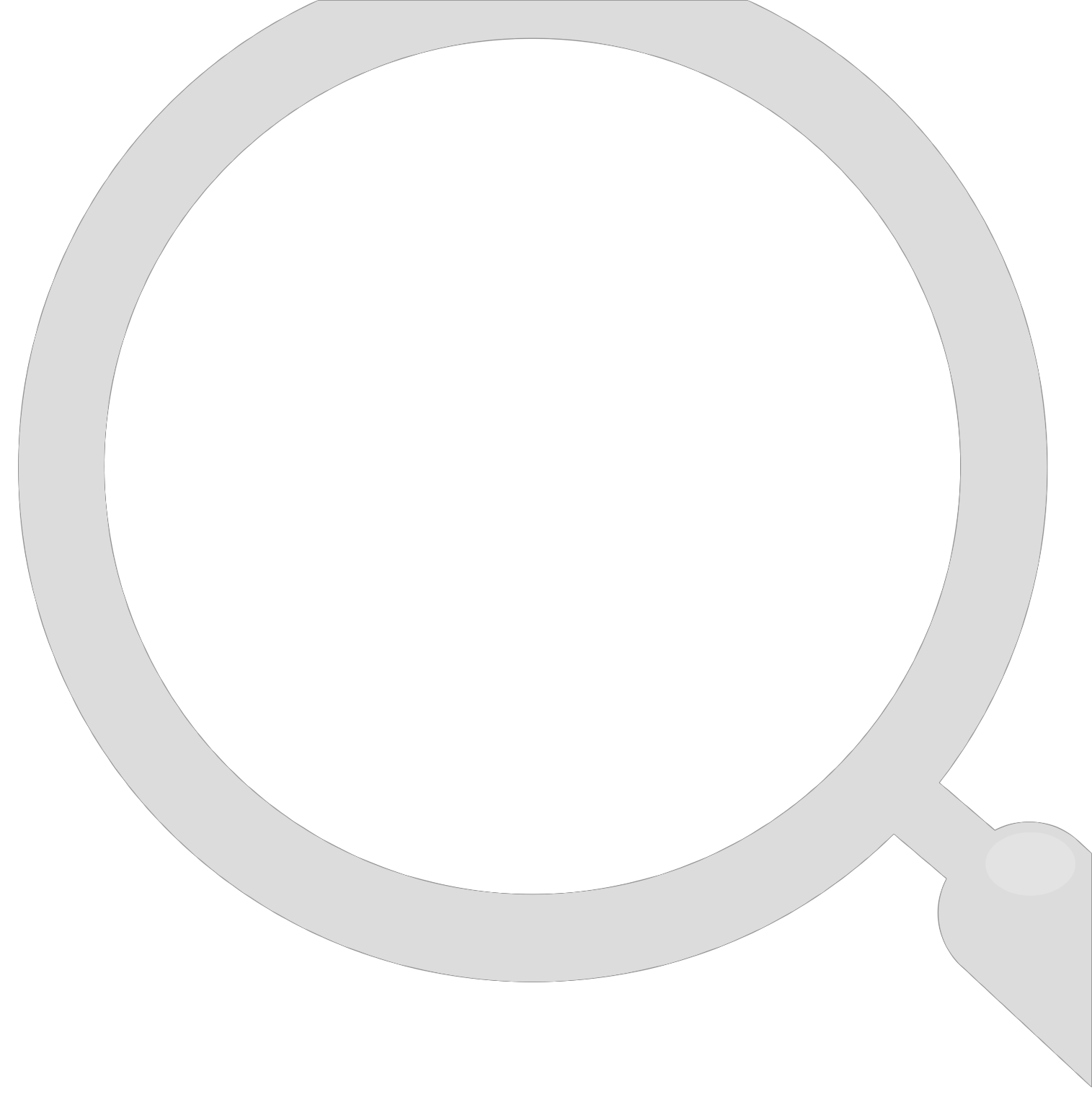
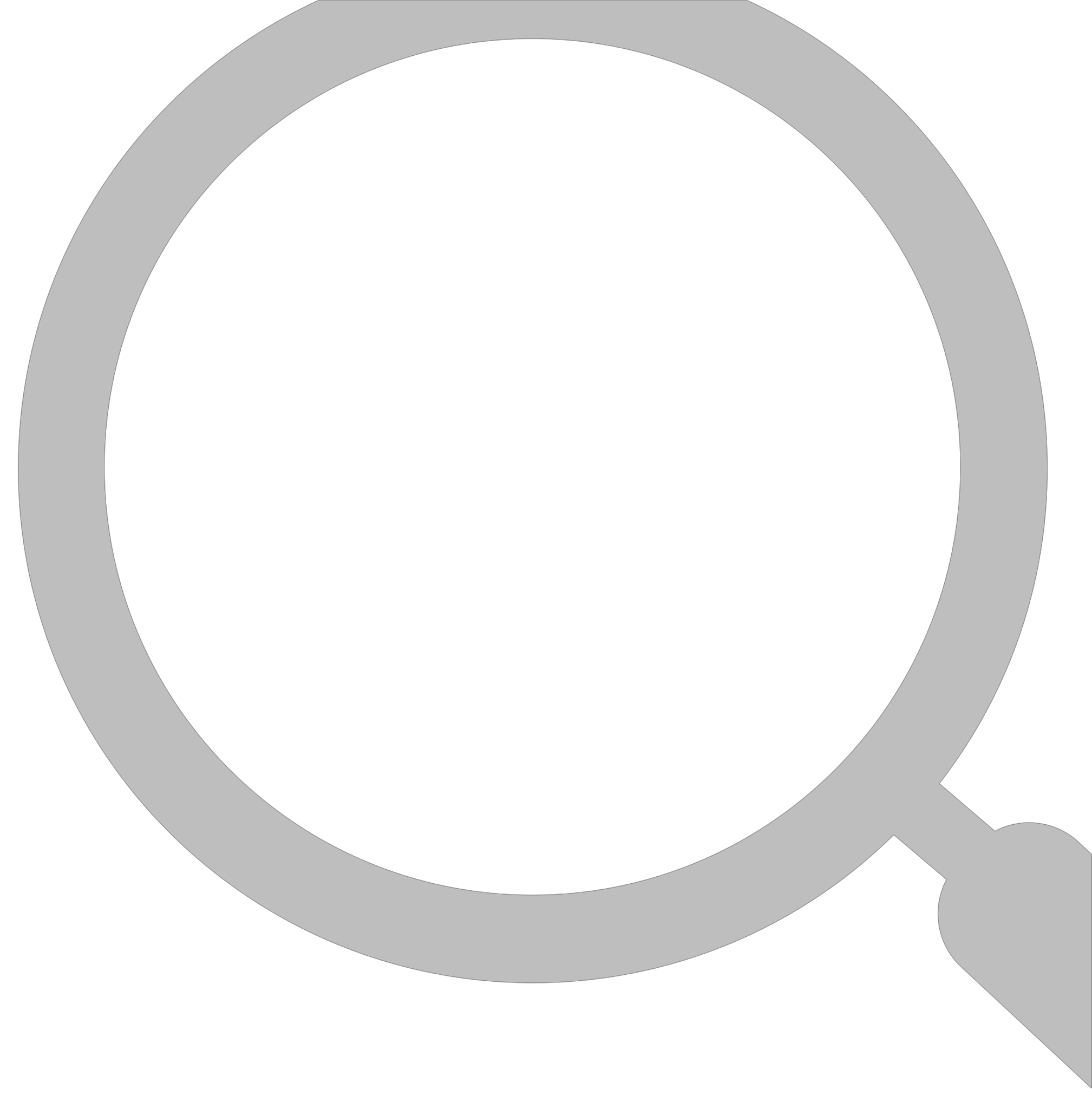
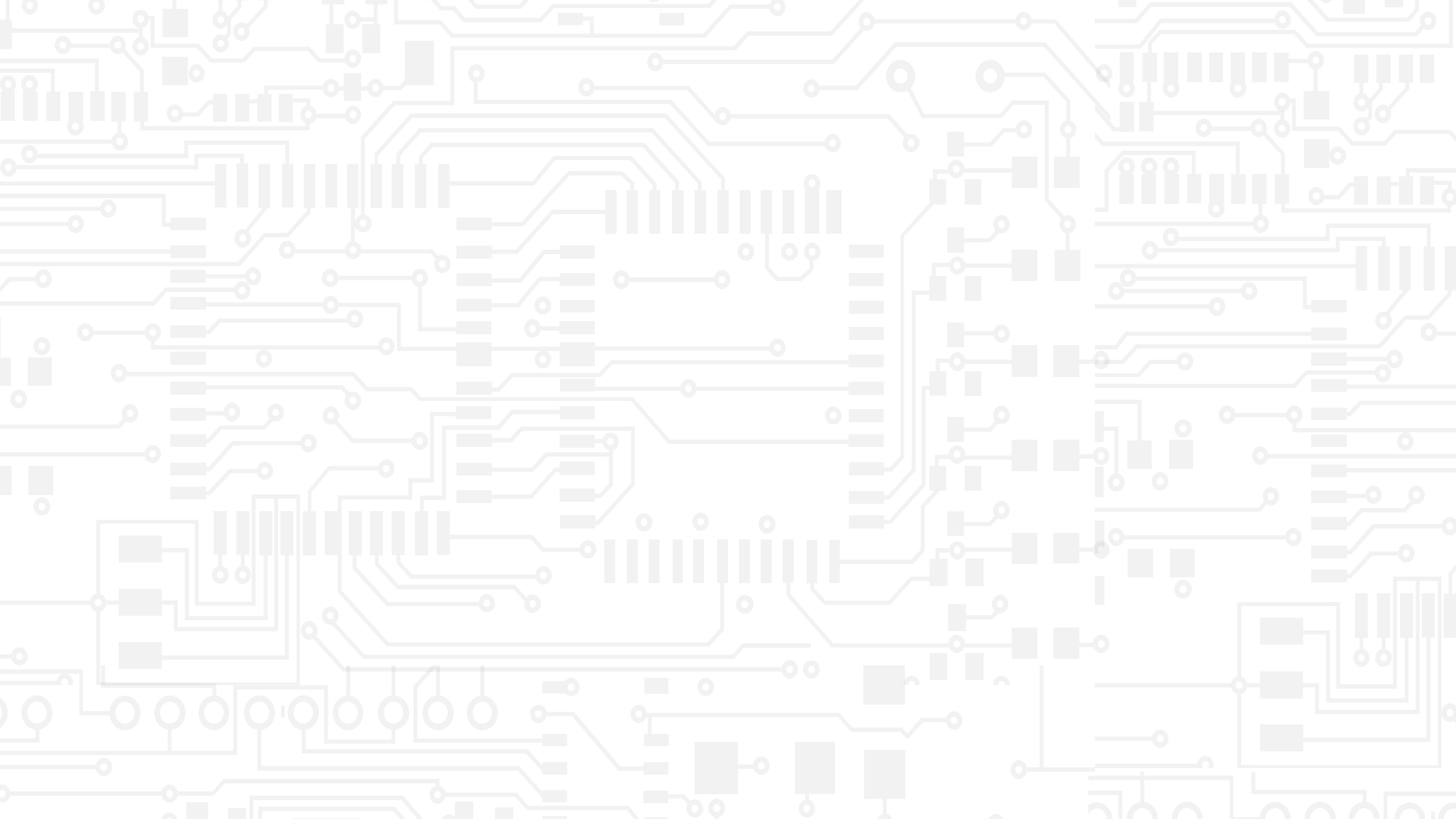
Python是跨平台语言

4

Python是多模型语言 **Guido van Rossum**

5

**Python语言创立者**



#### Python快速入门

**编译和解释**

编译执行

1

**源代码**

**编译器**

**目标代码**

## 将源代码一次性转换成目标代码的过程

**程序输入**

**程序执行**

编译执行

1

**源代码**

**编译器**

**目标代码**

**结果输出**

解释执行

2

**结果输出**

**解释器**

**源代码**

**程序输入**

## 将源代码逐条转换成目标代码同时逐条运行的过程

编译执行

1

解释执行

**结果输出**

**解释器**

**源代码**

**编译器**

**程序输入**

**程序执行**

2

**目标代码**

**结果输出**

**源代码**

**程序输入**

编译执行

1

解释执行

2

### 编译：一次性翻译，之后不再需要源代码（类似英文翻译）

* **解释：每次程序运行时随翻译随执行（类似实时的同声传译）**

静态语言：编译执行

2

1

脚本语言：解释执行

### 静态语言：C、C++、Java、Verilog …

* **脚本语言：PHP、JavaScript、HTML、Python …**

编译过程没有时间限制，优化更充分，执行速度更快

1

编译后程序可以在同系列系统中直接执行，不需要执行环境

2

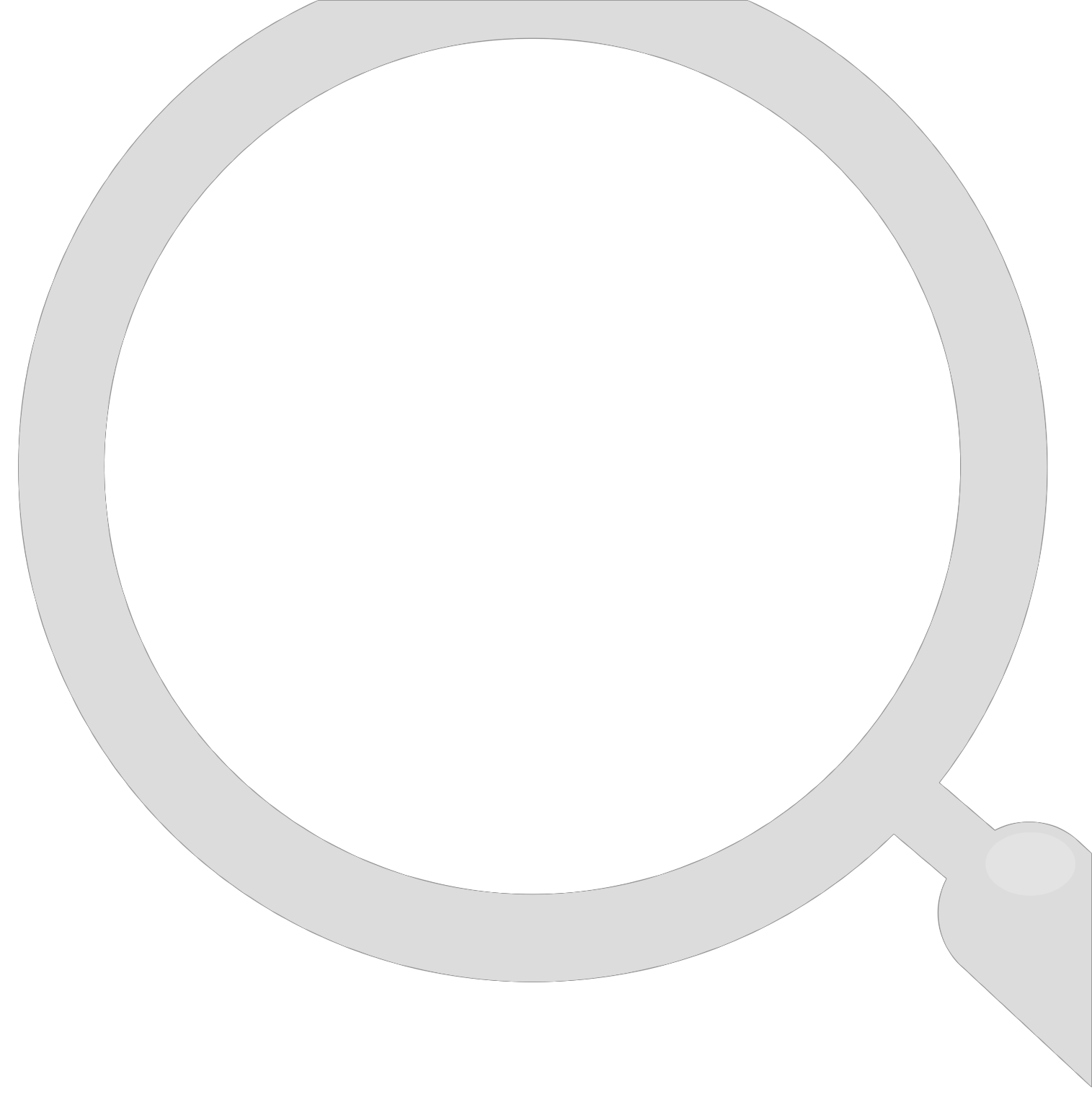
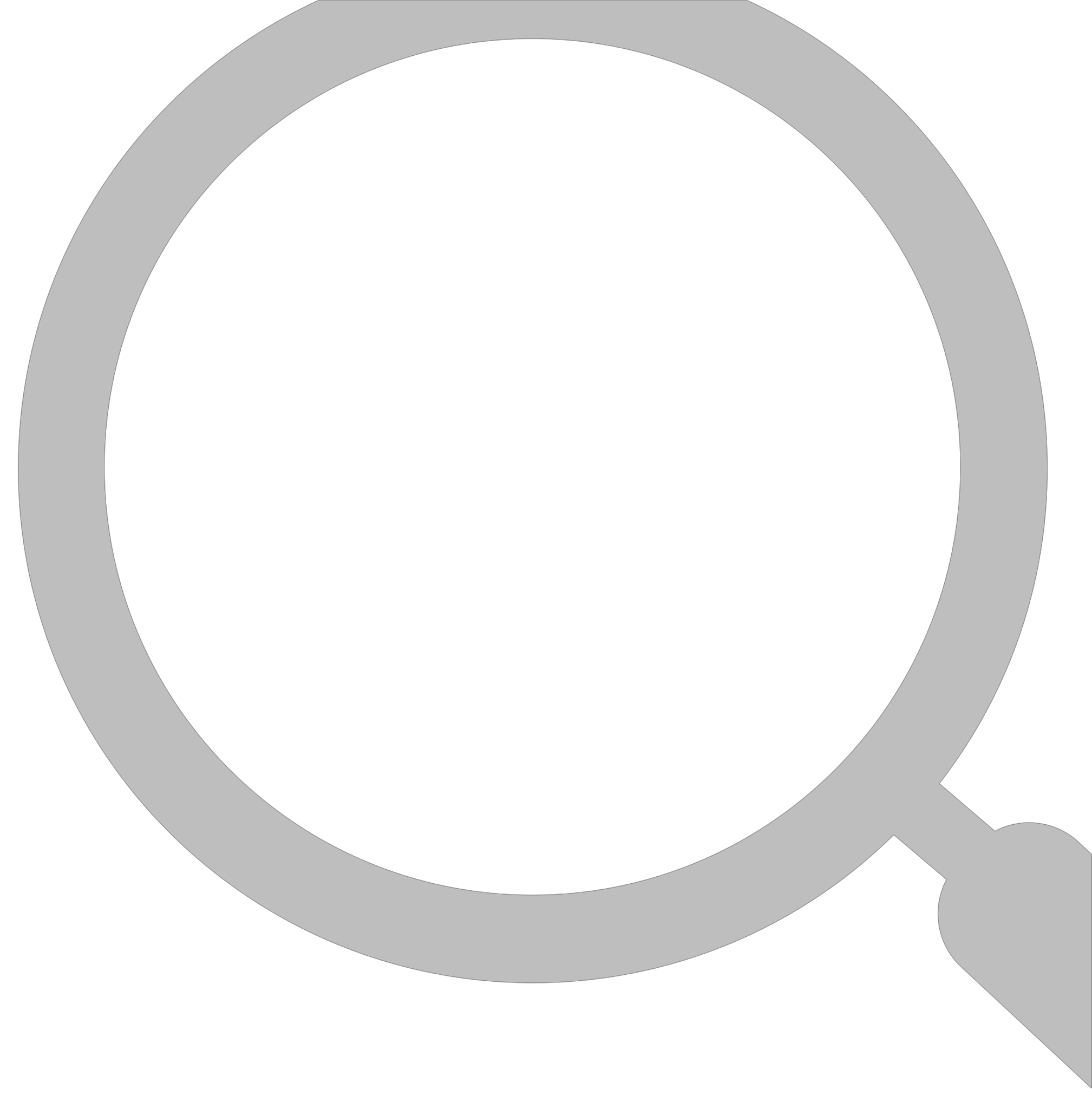
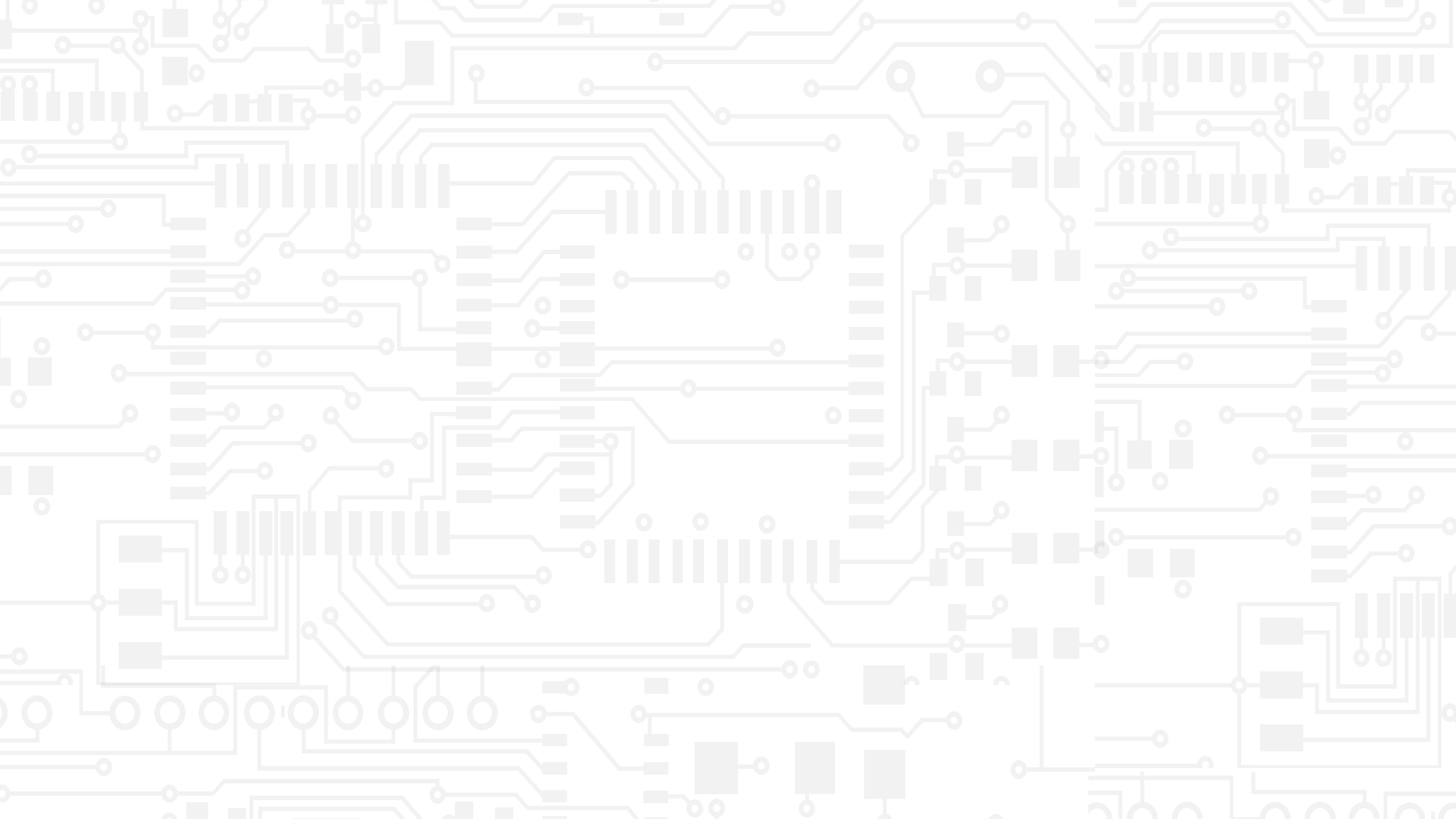
执行过程需要源代码，程序维护更灵活

1

通过在不同系统中配置执行环境，可以实现源代码的跨平台执行

2

## 编译和解释是两种方式的区别，在完成计算需求方面结果一致



#### Python快速入门

**Python语言特点**



**Python是通用语言**

1

Python是脚本语言

2

Python是胶水语言

3

Python是跨平台语言

4

Python是多模型语言 **Guido van Rossum**

5

**Python语言创立者**

##### 语法简洁

# 计算并输出1到100整数之和

s = 0

*for* i *in* range(1, 101): s += i

print(s)

# 计算并输出n的阶乘

*def* fact(*n*):

*if n* == 1:

*return* 1

*return n*\*fact(*n*-1) print(fact(10))

**语法简洁**

没有分号、没有函数约束、没有大括号、没有begin/end

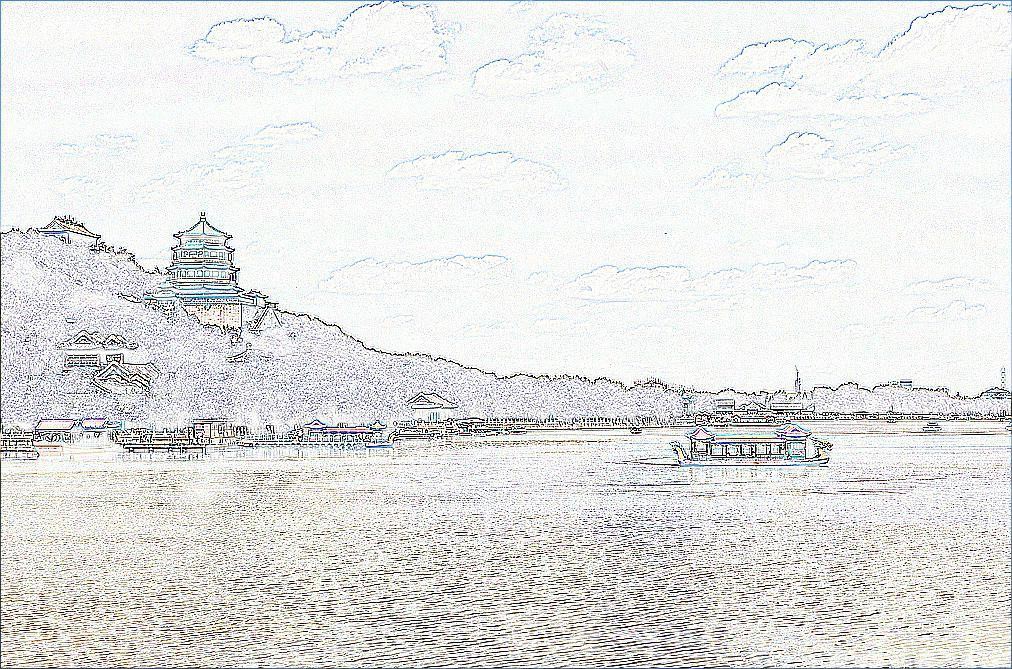
1

没有类型声明、没有变量定义、没有指针

2

## 相同功能是C语言代码量的10%及以下

##### 开源生态



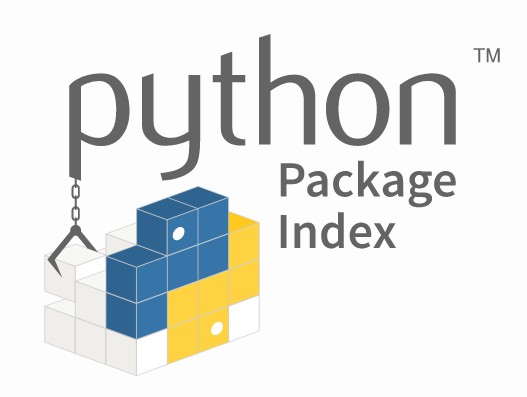
# 图像的轮廓获取

*from* PIL *import* Image

*from* PIL *import* ImageFilter im = Image.open('Beijing.jpg')

om = im.filter(ImageFilter.CONTOUR) om.save('BeijingContour.jpg')

**开源生态**

超过14万个第三方库免费可用避免重复造轮子的先进理念

1

2

## 底层可封装C/C++等语言的代码，站在巨人肩膀上编程

##### 语法简洁

**开源生态**

通过简化语法抽象逻辑，语法简洁能提高10倍左右的编程产量

1

通过功能模块复用集成，开源生态能再提高10倍左右的编程产量

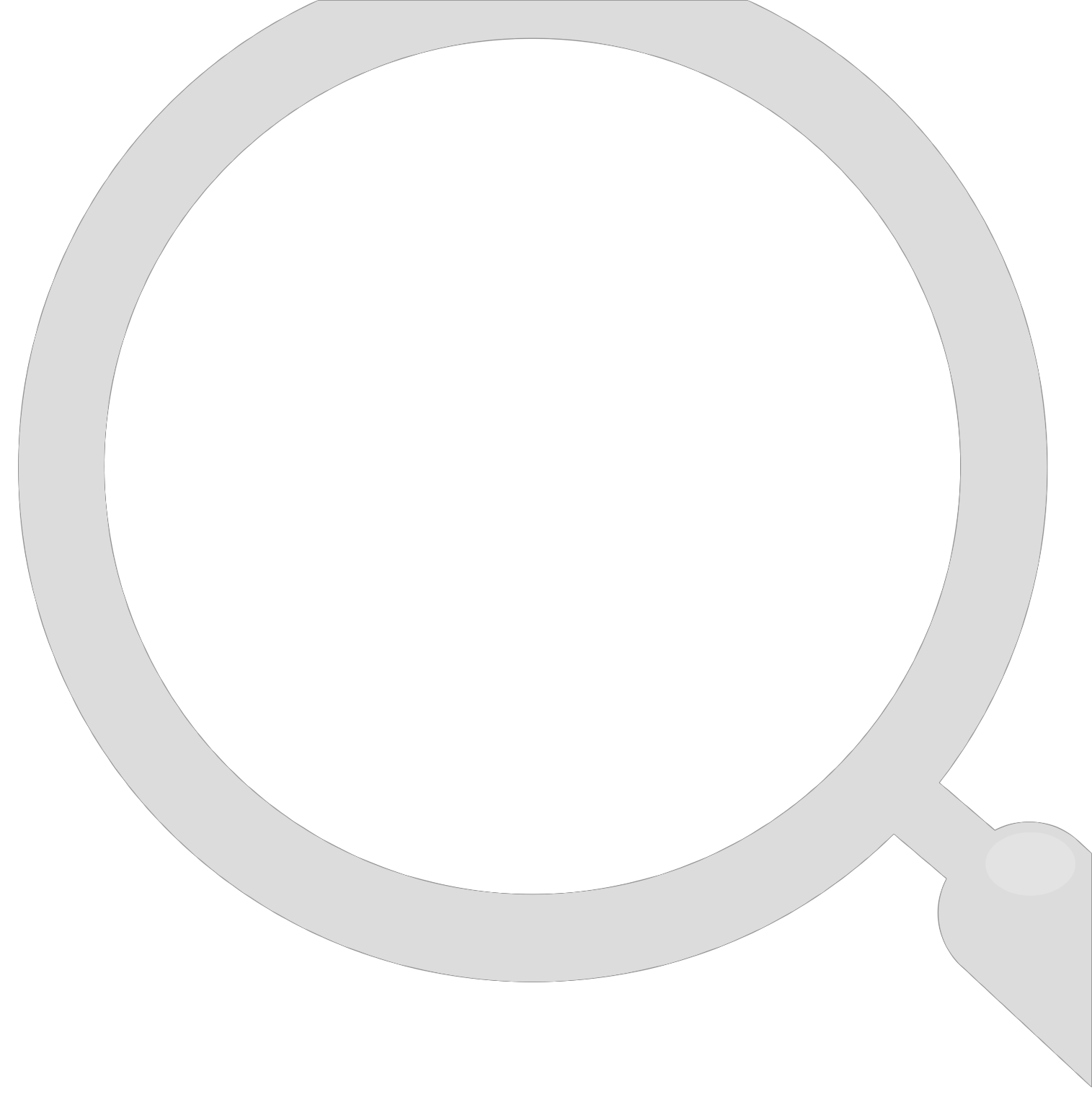
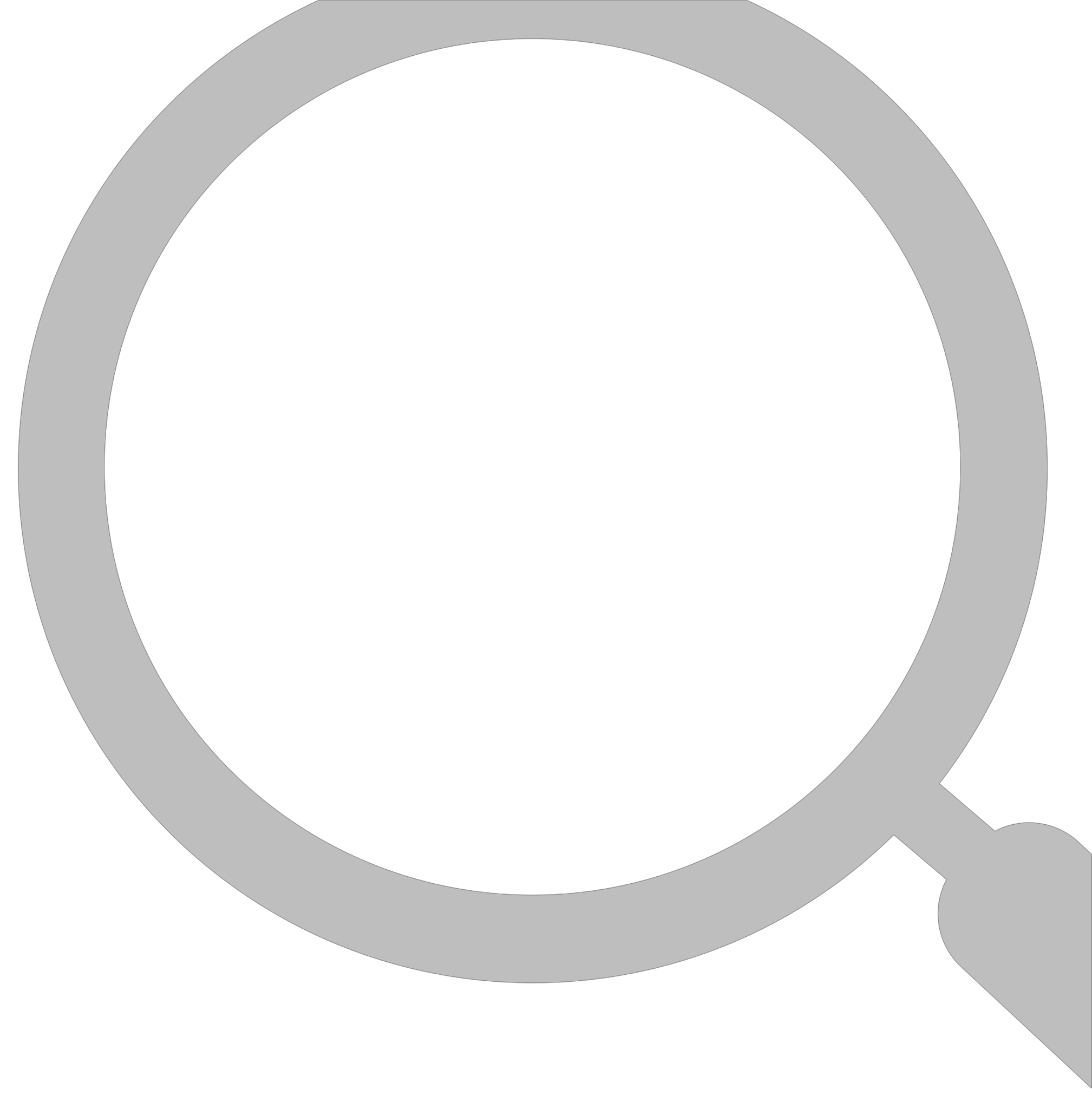
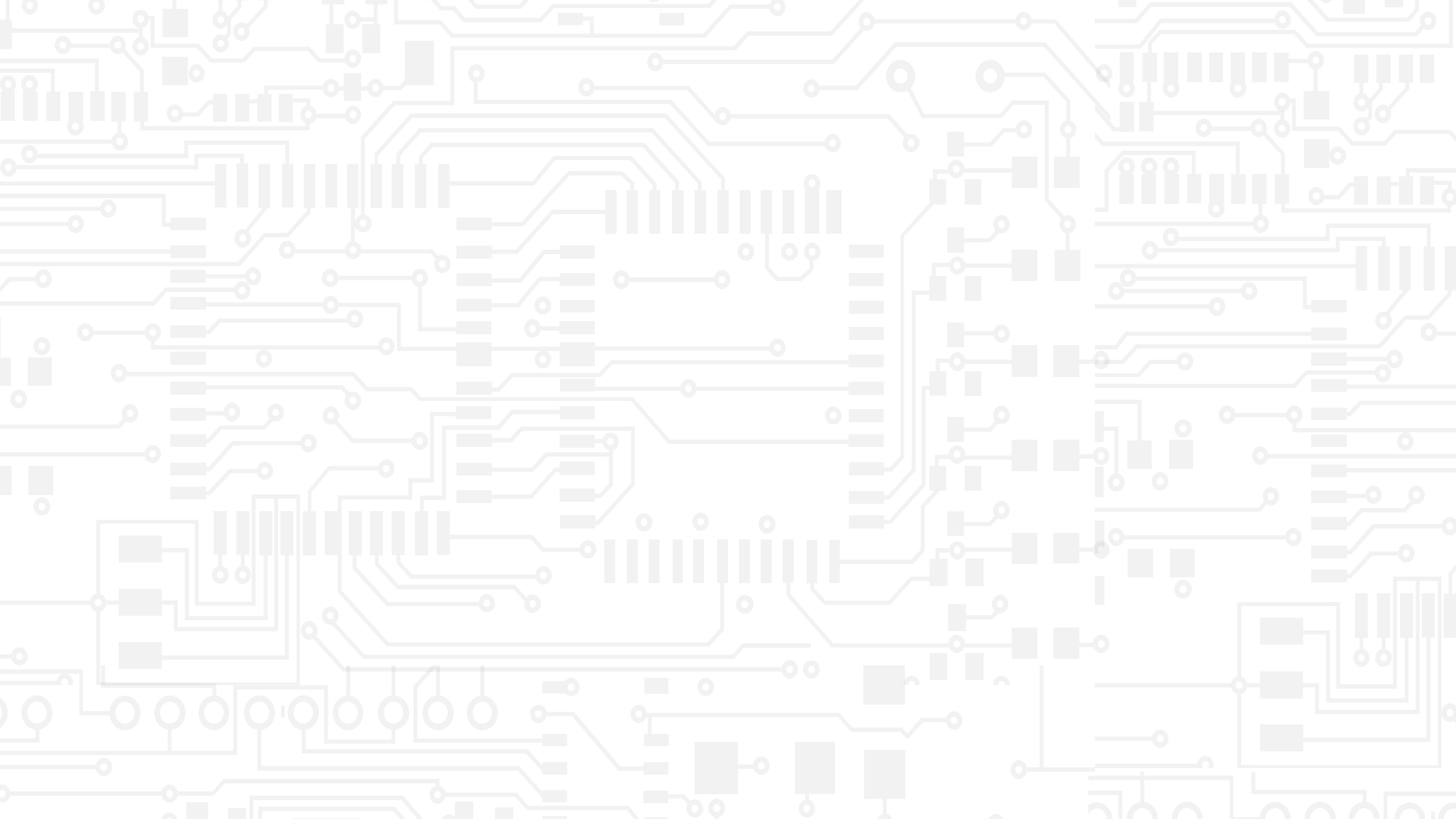
2

## Python能带来10倍以上的编程产量提升

##### 工具决定思维：关注工具变革的力量！



**>10x**



Python快速入门

**计算生态与超级语言**

**从开源运动说起…**

**Richard**

**Stallman启动**

**GNU项目**

**GNU通用许可协议诞生**

1983 1989

# 从开源运动说起…

##### Linus

**Torvalds发布了Linux内核**

**网景浏览器开源，产生了Mozilla**

1991 1998

1983, Richard Stallman

大教堂模式

**V.S.**

1991, Linus Torvalds

集市模式

##### 开源思想深入演化和发展，形成了计算生态

计算生态以开源项目为组织形式，充分利用“共 识原则”和“社会利他”组织人员，在竞争发展、相互依存和迅速更迭中完成信息技术的更新换代， 形成了技术的自我演化路径。



**没有顶层设计、以功能为单位、具备三个特点**

竞争发展相互依存迅速更迭

1

2

3

以开源项目为形式，提供了超过14万个第三方库

1

开源项目的建设经过野蛮生长和自然选择，高质量库众多第三方库之间相互关联使用，依存发展或逐级封装

2

3

4

Python社区庞大，新技术更迭迅速，已经成为计算生态主流

## API != 计算生态

**理解、运用和构建计算生态**

**编程语言的种类…**

##### 机器语言 汇编语言 高级语言

**11010010**

**00111011**

**add 2, 3, result result = 2 + 3**

##### CPU相关 CPU相关 CPU无关

**编程语言的种类…**

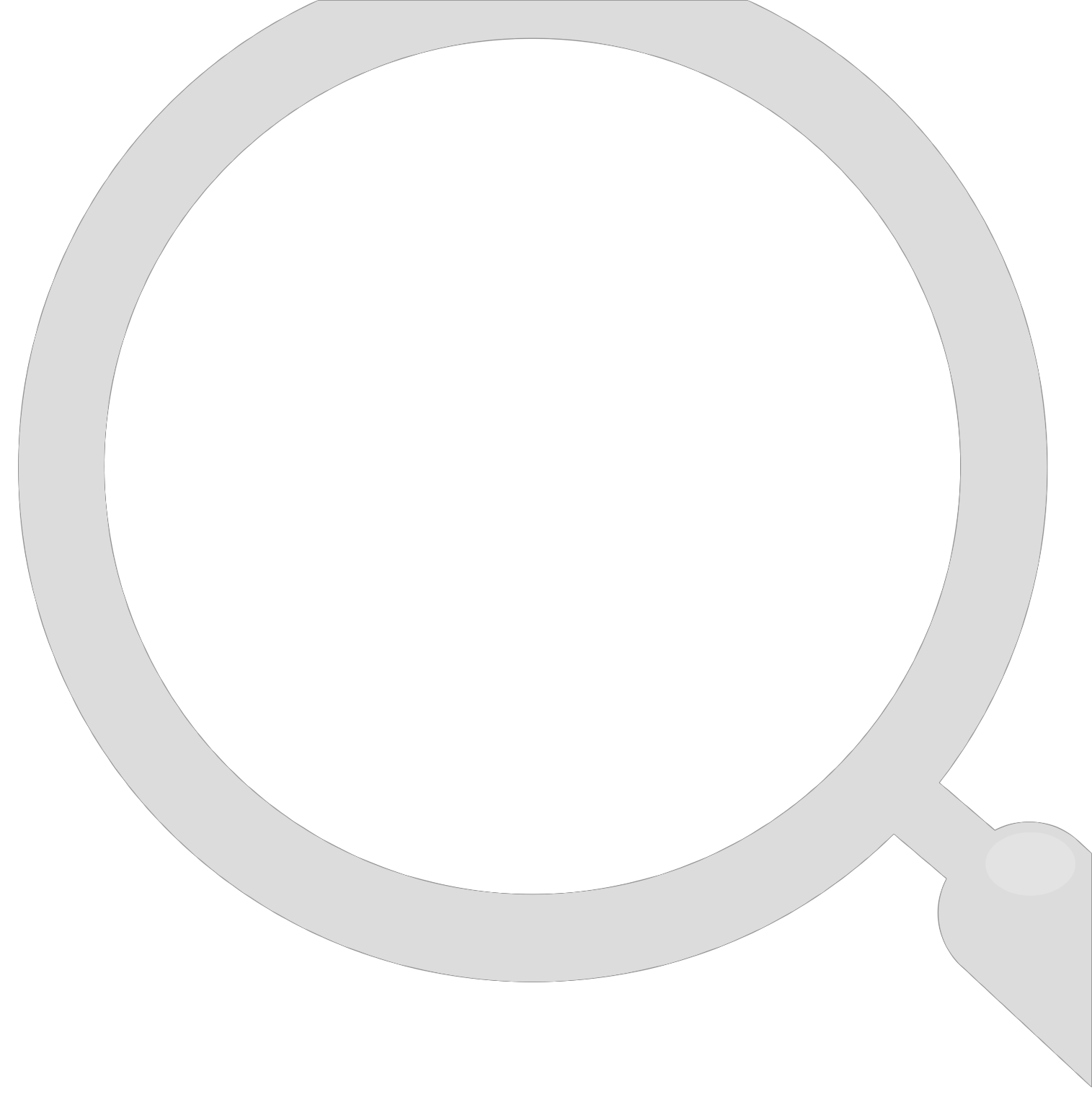
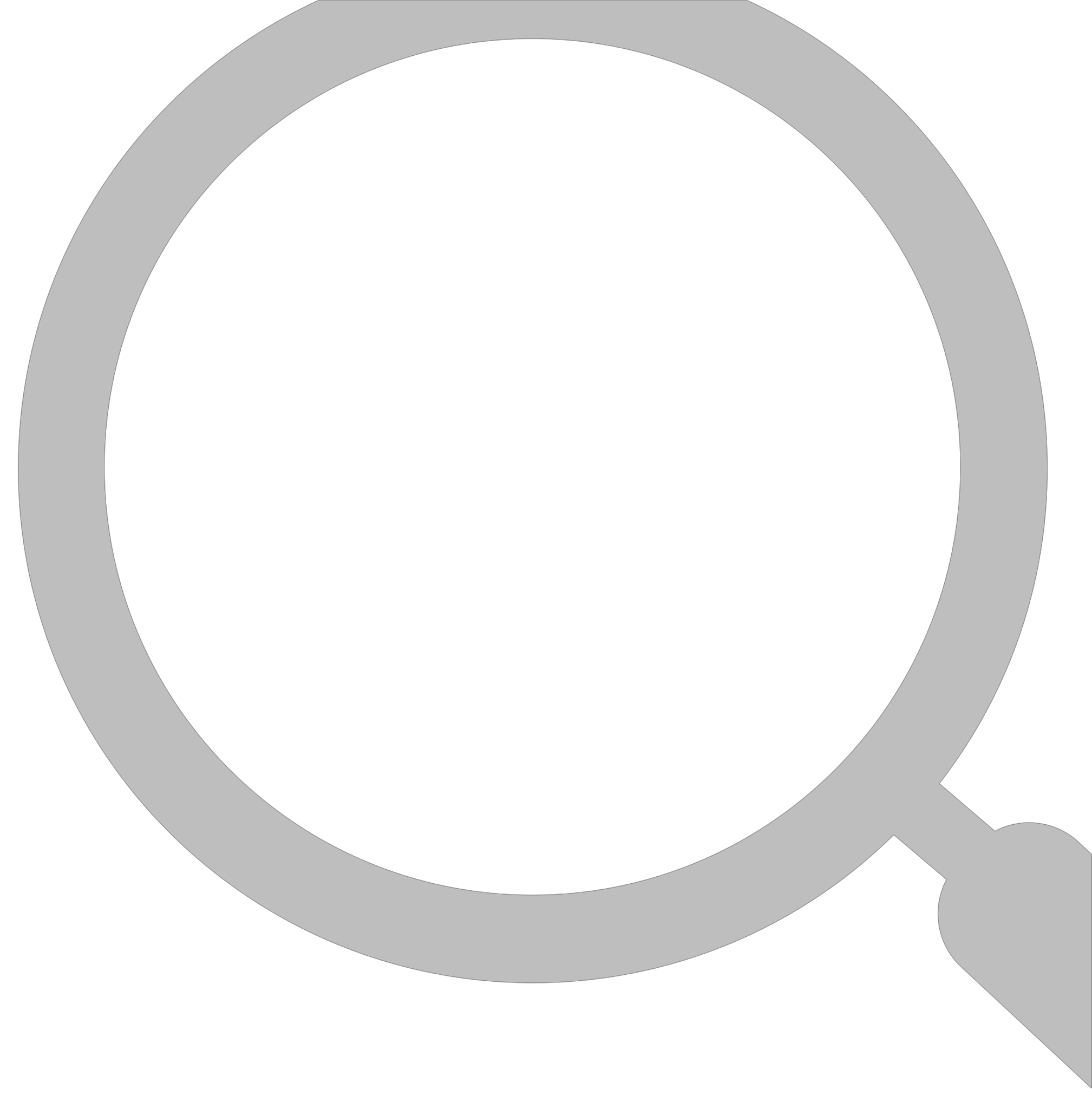
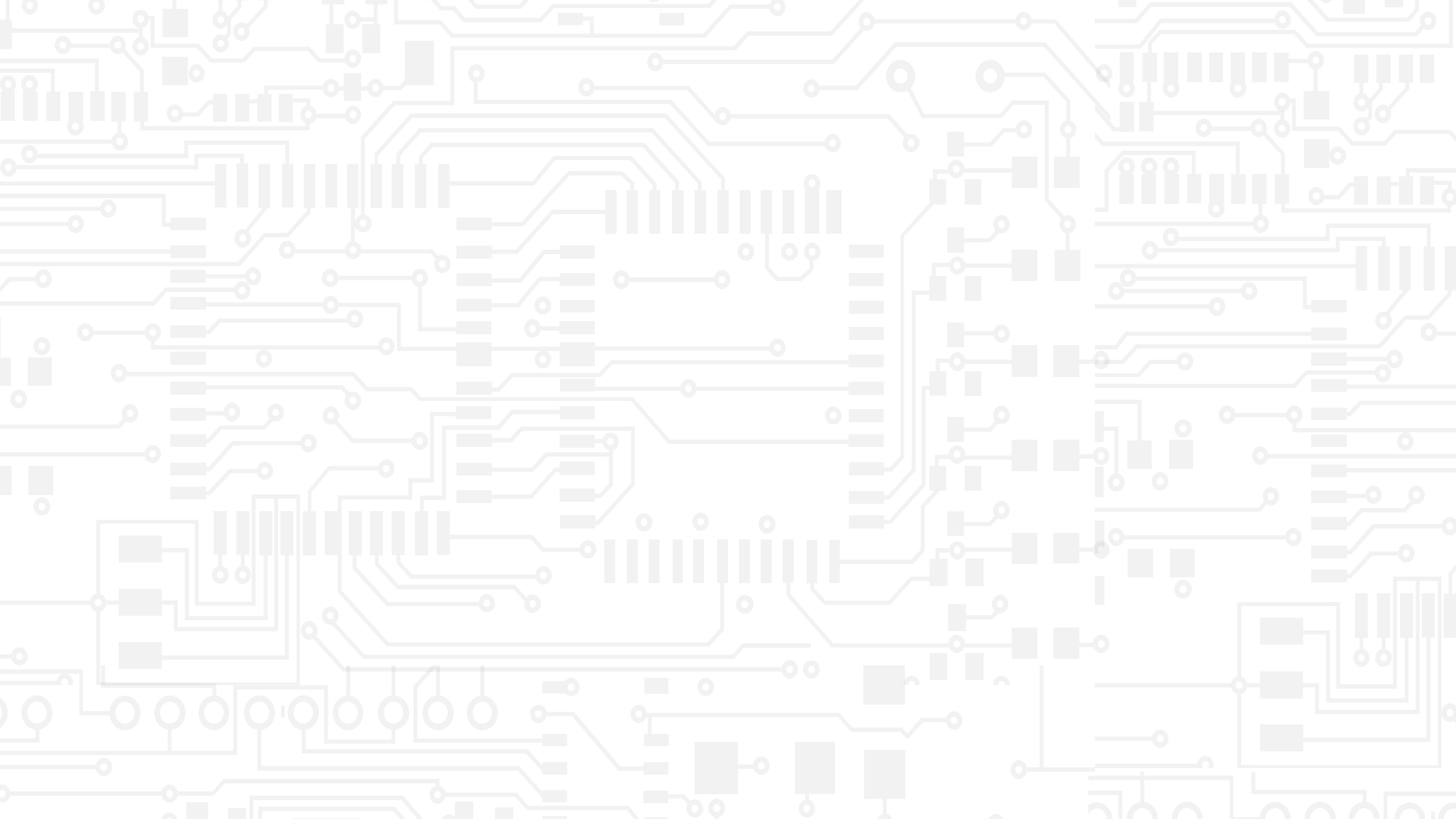
**汇编语言 高级语言 超级语言**

**add 2, 3, result result = 2 + 3**

##### CPU相关 CPU无关

**result = sum(2, 3)**

##### 计算生态



Python快速入门

**单元小结**

**单元小结**

**0**

**1**

**02**

**0**

**3**

**04**

计算机技术发展

与编程语言演进

编译与解释 Python语言特点 计算生态与超级语言



**Why Python？ What Python？ Thinking Python**

＼ *. \_/*

-一-- 一＿＿－－

\_\ /，

--------一 一一

／

．一-------

*------*

